

Subject Name: **Source Code Management**

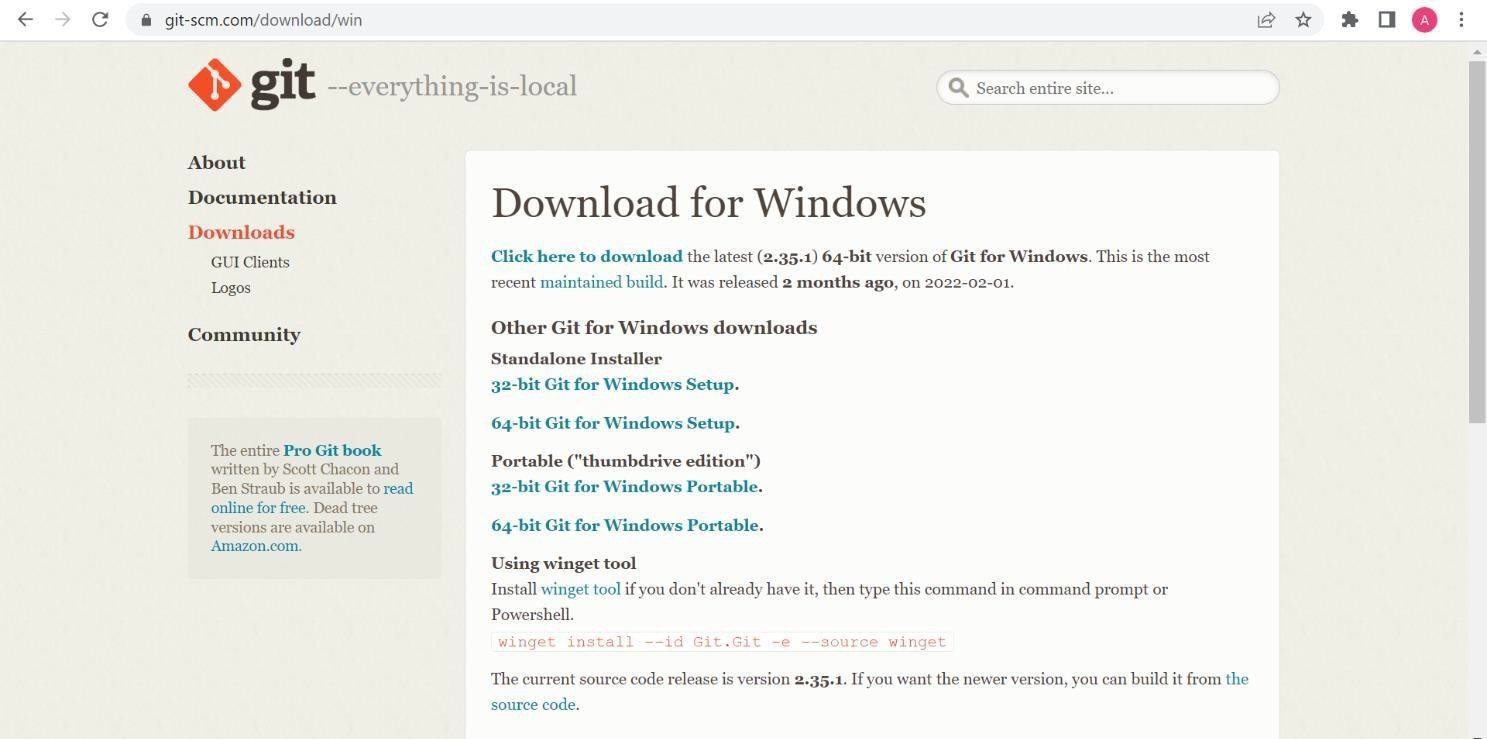
Subject Code: **24CSE0106** Session: **2024-25** Department: **CSE**

logo

|  |  |
| --- | --- |
| **Submitted By:**  Harshit Sharda 2410990331  G5 | **Submitted To:**  Dr.Chetna Sharma |

# List of Programs

|  |  |  |
| --- | --- | --- |
| **S. No** | **Program Title** | **Page No.** |
| 1 | Setting up of Git Client |  |
| 2 | Setting up GitHub Account |  |
| 3 | Generate logs |  |
| 4 | Create and visualize branches |  |
| 5 | Git life cycle description |  |
| 6 | Add Collaborators on GitHub Repository |  |
| 7 | Fork and Commit |  |
| 8 | Merge and Resolve conflicts created due to own activity |  |
| 9 | Reset and Revert |  |



## Practical 1

**Aim:**To install and configure Git Client on your local system.

## Theory:

Git is a distributed version control system used to track changes in source code. This practical focuses on setting up Git on your local system for effective version control.

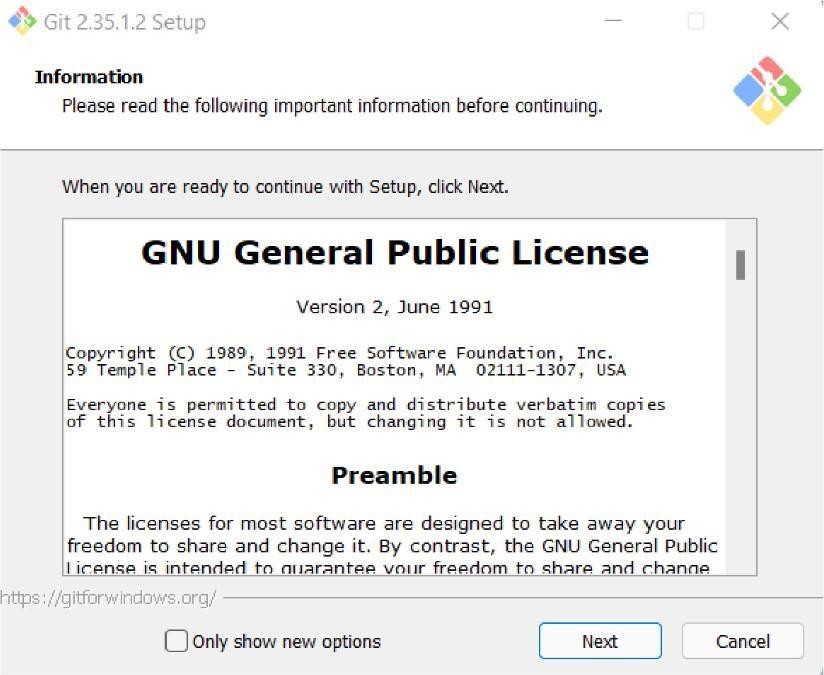
## Procedure:

* Download Git from git-scm.com.
* Install Git by following the setup wizard.
* Open Git Bash and verify installation using the command: git--version.
* Configure user details using the commands:

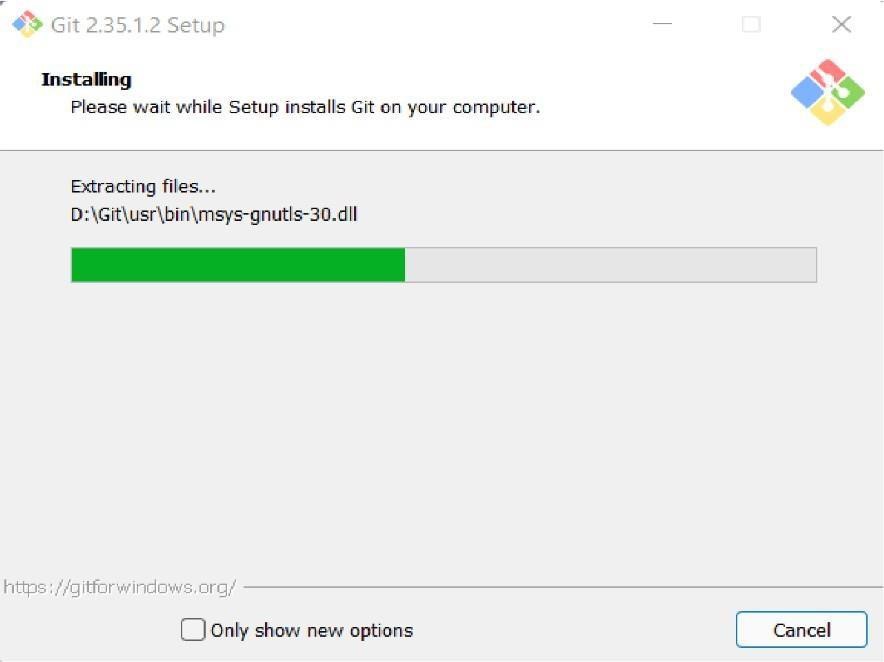
git config --global user.name "Your Name" git config--global user.email "Your Email"

## Snapshots of download:

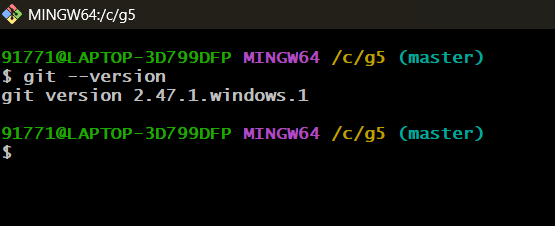
Opted for “64-bit Git for Windows Setup”



Git Setup



Git Installation



Git Bash version

## Practical 2

**Aim:**Setting up GitHub Account and Adding Collaborators on GitHub Repository

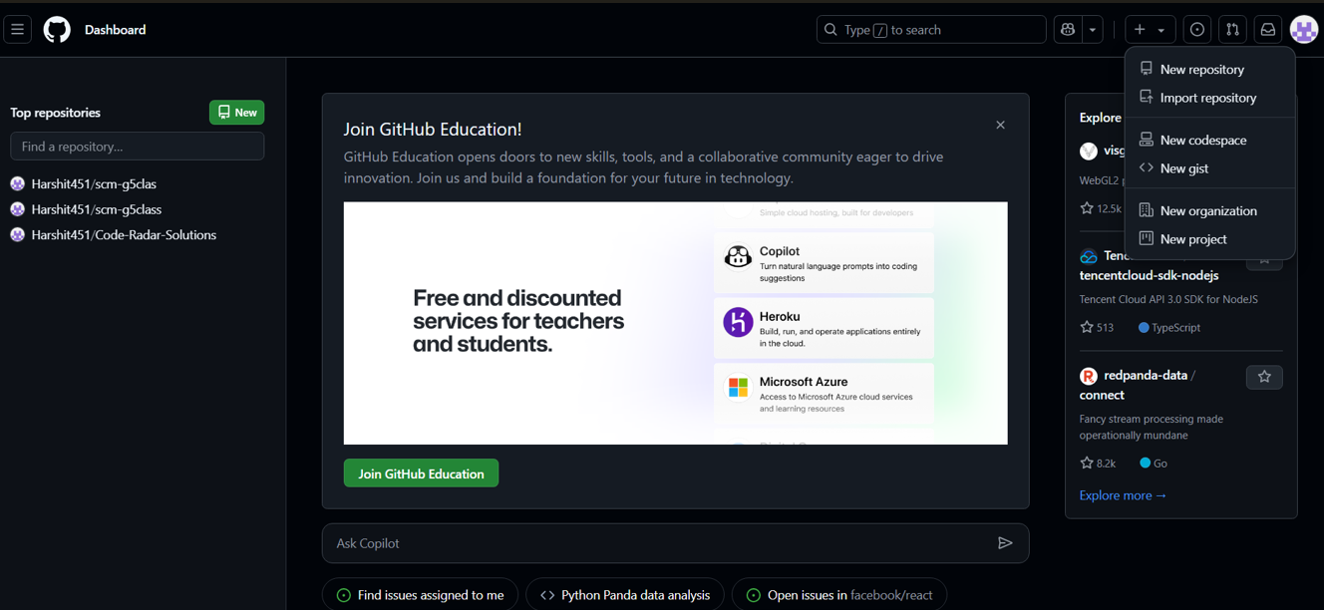
## Theory:

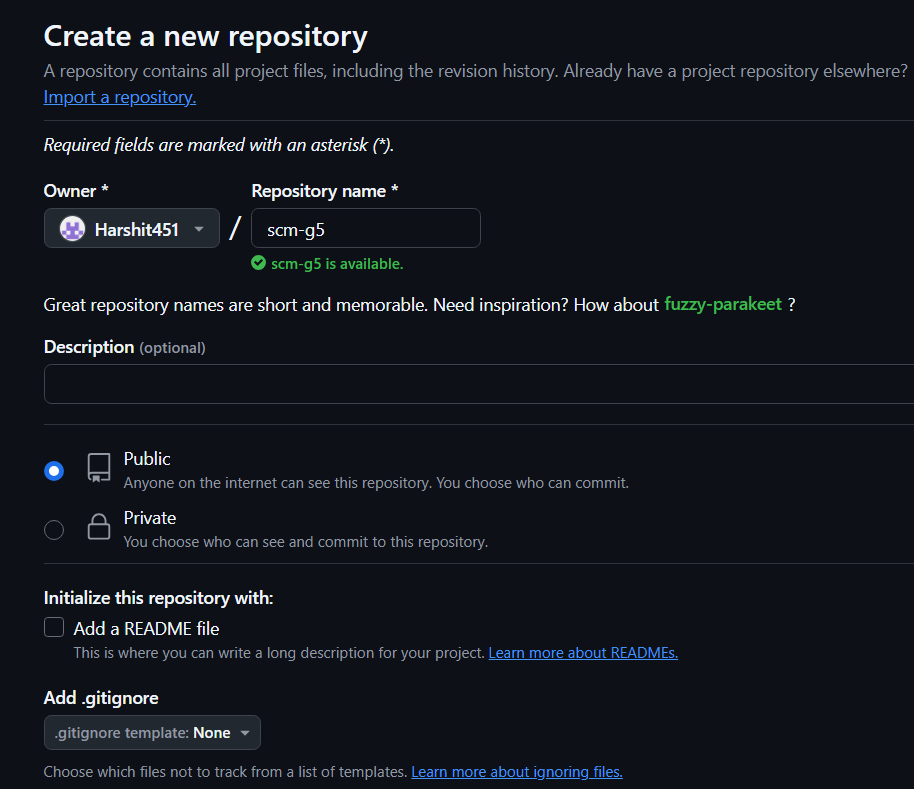
Collaborators are individuals with write access to a repository. They can contribute to the project by pushing changes and merging pull requests.

## Procedure:

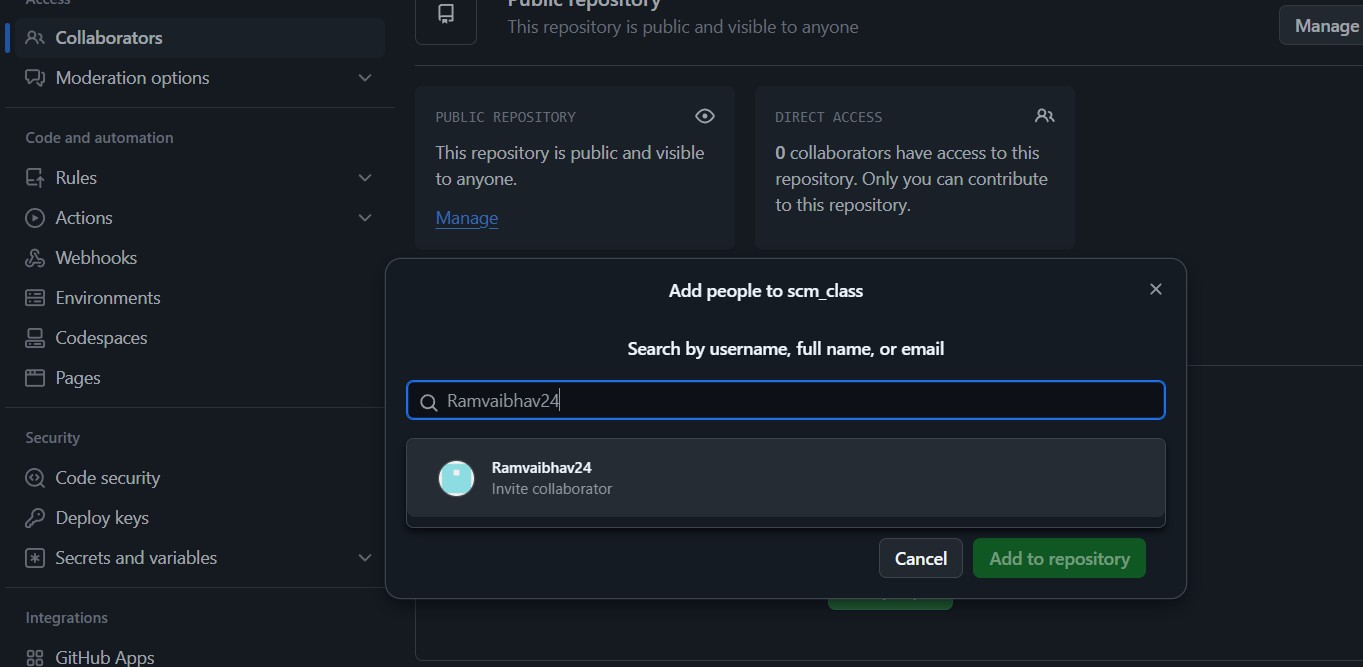
* Log in to your GitHub account and create a new repository.
* Navigate to Settings > Manage Access in the repository.
* Add collaborators by their GitHub usernames.
* Collaborators will receive an invitation email,which they must accept.

# Snapshots:









## Practical 3

**Aim:**

To merge two branches within a Git repository**.**

## Theory:

Merging branches in Git allows you to combine changes from one branch into another. It is a fundamental process in collaborative workflows, ensuring all contributions are integrated into a single codebase.

## Procedure:

1. Create a new branch and switch to it :

Git checkout-b new-branch

1. Make changes to a file in the newbranch and commit them: echo "New content"> file.txt

git add file.txt

git commit-m"Addchanges in new branch"

1. Switch back to the main branch :

git checkout main

1. Modify another file in the main branch and commit the changes: echo "Main branch changes"> another-file.txt

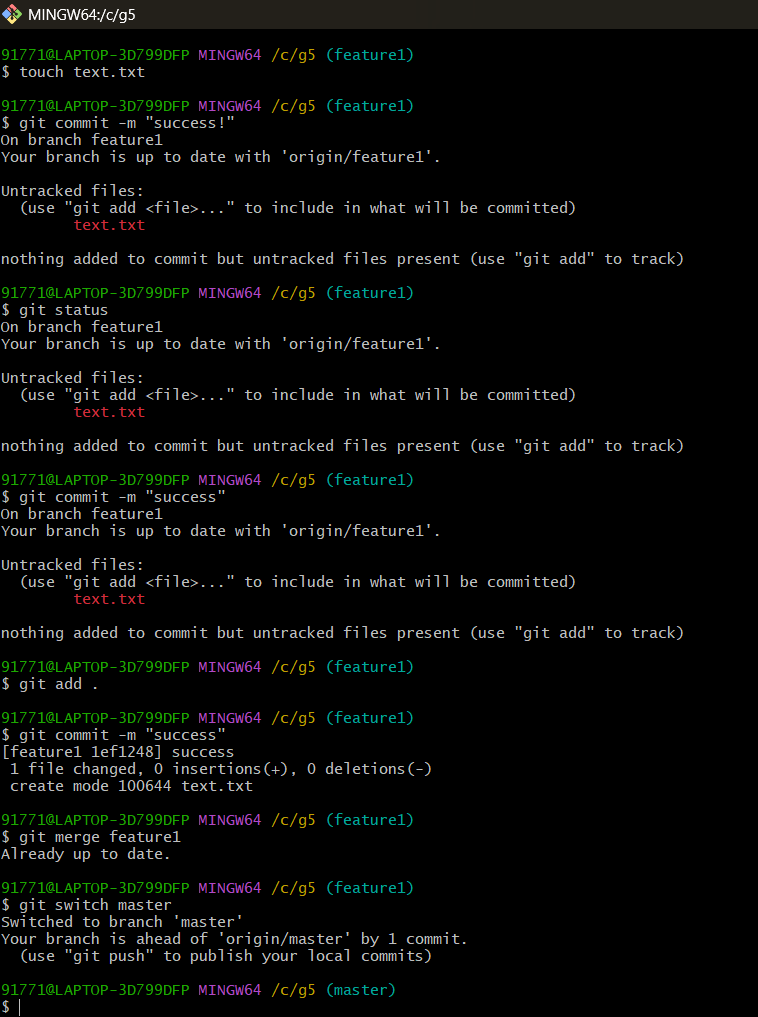
git add another-file.txt

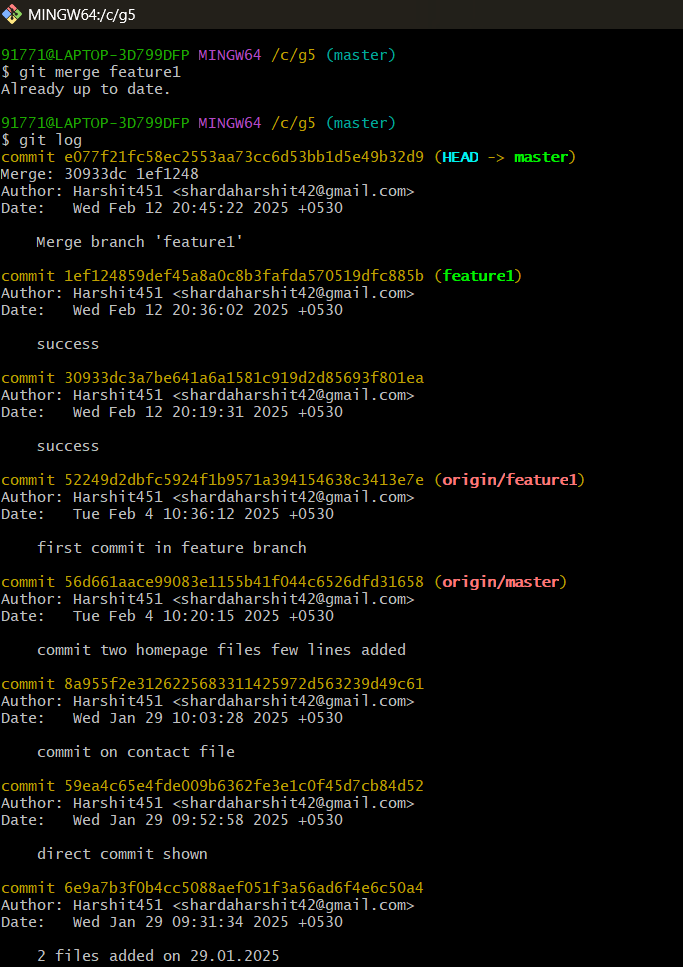
git commit-m"Modify file in main branch"

1. Merge the new branch into the main branch :

Git merge new-branch git merge new-branch

**Snapshots:**





## Practical 4 Aim:

To demonstrate push and pull operations in Git.

## Theory:

Push transfers committed changes from the local repository to the remote repository, while pull retrieves updates from the remote repository.

## Procedure:

* Make changes in the local repository and commi them.
* Push the changes to the remote repository using git push.
* Make changes directly on the remote repository (e.g.,via GitHub interface).
* Pull the changes to the local repository using git pull.

## Tasks:

Provide screenshots of the push and pull operations. Include the updated commit log.

**Screenshots**

